The Calculation of the Mass Moment of Inertia of a Fluid in a Rotating Rectangular Tank

124

SUMMARY

This analysis calculated the mass moment of inertia of a non-viscous fluid in a slowly rotating rectangular tank. Given the dimensions of the tank in the x, y, and z coordinates, the axis of rotation, the percentage of the tank occupied by the fluid, and angle of rotation, an algorithm was written that could calculate the mass moment of inertia of the fluid. While not included in this paper, the change in the mass moment of inertia of the fluid could then be used to calculate the force exerted by the fluid on the container wall.

ANALYSIS

Depending on the dimensions of the tank, the amount of fluid in the tank, and the angle of rotation, the resulting shape of the fluid can be broken down into simple geometries. Since the fluid was assumed to be incompressible, non-viscous, and the rotational velocity small, the shape of the fluid was represented by rectangular and triangular prisms. The most simple fluid shapes were represented by a single rectangular or a single triangular prism, while the most complex geometry was represented by two rectangular and one triangular prism (See Fig. 1 and 2). The mass moment of inertia ("Iz" in this coordinate system) of each prism was first calculated relative to its centroid. Then the parallel axis theorem was used to calculate the Iz of the entire volume at either the fluid's center of mass or the container's center of rotation.

(NASA-CR-197777) THE CALCULATION OF THE MASS MOMENT OF INERTIA OF A FLUID IN A ROTATING RECTANGULAR TANK Final Report (California State Univ.) 46 p

$$\overline{I_z} = \frac{m(a^2 + b^2)}{12}$$

where:

I, = Moment of inertia with respect to the z-axis.

 a^* = Length of rectangular prism in x-axis.

b = Length of rectangular prism in y-axis.

m = Mass of prism.

The mass moment of inertia of a triangular prism with respect to its centroid and the axis shown was calculated by the following equations: 2

$$\overline{I_z} = \frac{m(a^2 + b^2)}{18}$$

The parallel axis theorem: 1

$$I_z - \overline{I_z} + m(\overline{x^2} + \overline{y^2})$$

where:

x = Distance in x-direction from centroidal to arbitrary axis.

y = Distance in y-direction from centroidal to arbitrary axis.

For comparison, the effective moment of inertia of fluid was calculated by the following formula:3

$$\frac{I_{ry}}{I_{Sy}} = 1 - \frac{4r_1^2}{1 + r_1^2} + 2.510 \left(\tanh \frac{\pi}{2r_1} + 0.0045 \right) \left(\frac{r_1^3}{1 + r_1^2} \right)$$

where:

 I_{FZ} = Effective moment of inertia of fuel about z-axis. I_{SZ} = Moment of inertia of solidified fuel about z-axis.

= Iz when theta equals zero.

r1 = b/a = tank aspect ratio in xy-plane.

RESULTS

As the partially filled rectangular tank shown in figures 1 or 2 rotates about its origin, the Iz of the fluid will change. Program MomentOfInertia (See Appendix B) calculated the Iz of the fluid relative to the centroid of the fluid. In tables one through three, three different tank dimensions (all with unit depth) are shown: 1×2 , 1×4 , and 1×8 , respectively. With each tank dimension, the mass moment of inertia for three different fluid volumes were tabulated. Only zero through 90 degrees were calculated since the mass moment of inertia for 90 through 180 degrees are mirror image of the shown data. The data then repeats every 180 degrees. This data is also shown graphically in figures 3 through 6.

As expected, the Iz of tanks with aspect ratio of 1/2 did not change significantly relative to tank rotation at any fluid level. This was due to the proximity of the aspect ratio to unity. With smaller (or greater) aspect ratios, the change in Iz increased significantly. For 50% volume, there was a 112% increase in Iz/rho for a/b = 1/2; while for a/b = 1/8, the change was 278%. Decreasing fluid volume also increased the change in Iz as the tank rotated. For a/b = 1/8, and the fluid volume was 80 percent, the change in Iz was 54%. For 20 percent fluid volume and the same aspect ratio, the change in Iz was 1,700%.

Program Moment1 (See Appendix B) calculated the Iz of the fluid relative to the center of rotation, which in this case was the origin. The calculations were very similar to the previous, except that the center of rotation was used as the axis instead of the C.G. of the fluid. Tables 3 through 6 shown the values calculated for 3 different aspect ratios and 3 different fluid volumes for each aspect ratio. These values are shown graphically in figures 7 thru 9.

As compared to the Iz relative to the C.G. of the fluid, the Iz relative to the center of rotation was greater, given the same aspect ratio and fluid volume. The greatest increase was the 20 percent fluid volume and aspect ratio = 1/2. The smaller percentage filled containers had greater increases than the higher percentage filled containers due to the greater change in the distance between the C.G. of the fluid and the center of rotation. For example, while the Iz of zero degrees, aspect ratio = 1/5, and 80 percent fluid volume changed 17% when the axis changed from C.G. of fluid to center of rotation, the Iz for the same angle and

aspect ratio, but only 20% fluid volume, changed 1,873%.

The change in the mass moment of inertia also increased with decreasing aspect ratio and decreasing fluid volume, however, the percent change was not as dramatic. The change in Iz for aspect ratio = 1/2 and 20% fluid volume relative to the C.G. was 248%, while the change in Iz relative to center of rotation was reduced to 48%.

The Iz of a tank with decreasing fluid level was also calculated by altering program MomentCG. The fluid level started at 100% at 0 degrees and decreased to 0% after 360 degrees. The results for three different aspect ratios are plotted in Figures 7 thru 9. This data demonstrated that even with an aspect ratio = 1/2, the change in Iz can be significant when the fluid level decreased. With even smaller aspect ratios, such as wing fuel tanks, the change in Iz was even greater. The "jumps" in the value of Iz corresponds to the a large change in the geometry of the fluid.

For comparison, the effective moment of inertia for various tank aspect ratios were also calculated. The effective moment of inertia is the moment of inertia of an equivalent mechanical system. However, the effective moment applies to small angular displacements only, which was quite different from the case that was analyzed here. These values are plotted in Figures 10 thru 12.

REFERENCES

- 1. Beer, F.P., and Johnston E.R., <u>Vector Mechanics for Engineers</u>, Dynamics, McGraw-Hill Book Co., New York, 1977.
- 2. Higdon and Stiles, <u>Engineering Mechanics</u>, Prentice-Hall, New Jersey, 1968.
- 3. Graham, E.W., and Rodriguez, A.M., "The characteristics of Fuel Motion Which Affect Airplane Dynamics", <u>Journal of Applied Mechanics</u>, September 1952.

APPENDIX A TABLES AND FIGURES

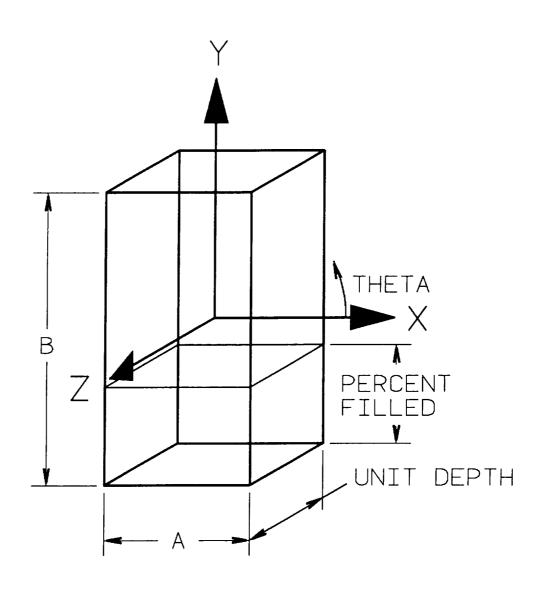


FIGURE 1: TANK CONFIGURATION

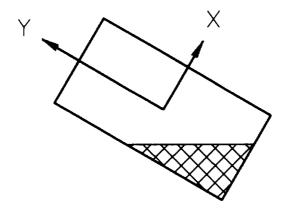


FIGURE 2: MOST SIMPLE GEOMETRY.

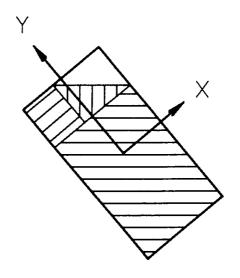


FIGURE 3: MOST COMPLEX GEOMETRY.

Aspect Ratio = 1/2

Theta	Izz/rho			
	20% Full	50% Full	80% Full	
0.0	.0387	.1667	.4747	
10.0	.0386	.1677	.4766	
20.0	.0385	.1712	.4829	
30.0	.0380	.1780	.4953	
40.0	.0361	.1903	.5178	
50.0	.0361	.2125	.5440	
60.0	.0411	.2552	.5703	
70.0	.0553	.3154	.5989	
80.0	.1019	.3454	.6183	
90.0	.1347	.3542	.6187	
% Change	248	112	30	

Table 1: Iz relative to fluid center of gravity for tank dimensions: a = 1.00, b = 2.00, and c = 1.00.

Aspect Ratio = 1/5

Theta	Izz/rho			
	20% Full	50% Full	80% Full	
0.0	.1093	.8333	2.9973	
10.0	.1101	.8358	3.0014	
20.0	.1126	.8439	3.0147	
30.0	.1173	.8599	3.0410	
40.0	.1256	.8891	3.0894	
50.0	.1400	.9450	3.1825	
60.0	.1642	1.0651	3.3819	
70.0	.2213	1.3867	3.6922	
80.0	.4158	2.3042	4.1547	
90.0	1.0693	2.7083	4.4373	
% Change	878	225	48	

Table 2: Iz relative to fluid center of gravity for tank dimensions: a = 1.00, b = 4.00, and c = 1.00.

Aspect Ratio = 1/8

Theta	Izz/rho			
	20% Full	50% Full	80% Full	
0.0	.4747	5.6667	22.378	
10.0	.4766	5.6718	22.387	
20.0	.4829	5.6885	22.467	
30.0	.4953	5.7216	22.467	
40.0	.5180	5.7826	22.566	
50.0	.5610	5.9000	22.755	
60.0	.6519	6.1581	23.173	
70.0	.8833	6.8869	24.368	
80.0	1.6633	10.522	28.232	
90.0	8.5387	21.417	34.475	
% Change	1699	278	54	

Table 3: Iz relative to fluid center of gravity for tank dimensions: a = 1.00, b = 8.00, and c = 1.00.

Aspect Ratio = 1/2

Theta	Izz/rho			
	20% Full	50% Full	80% Full	
0.0	.2947	.4167	.5387	
10.0	.2931	.4167	.5390	
20.0	.2880	.4167	.5453	
30.0	.2780	.4167	.5553	
40.0	.2596	.4167	.5737	
50.0	.2387	.4167	.5946	
60.0	.2187	.4167	.6147	
70.0	.1986	.4167	.6347	
80.0	.1924	.4167	.6409	
90.0	.1987	.4167	.6347	
% Change	34	0	18	

Table 4: Iz relative to center of rotation for tank dimensions: a = 1.00, b = 2.00, and c = 1.00.

Aspect Ratio = 1/5

Theta	Iz/rho				
	20% Full	50% Full	80% Full		
0.0	2.1573	2.8333	3.5093		
10.0	2.1542	2.8333	3.5124		
20.0	2.1441	2.8333	3.5226		
30.0	2.1240	2.8333	3.5427		
40.0	2.0869	2.8333	3.5797		
50.0	2.0153	2.8333	3.6514		
60.0	1.8067	2.8333	3.8060		
70.0	1.6238	2.8333	4.0428		
80.0	1.2927	2.8333	4.3740		
90.0	1.1973	2.8333	4.4693		
% Change	44	0	27		

Table 5: Iz relative to center of rotation for tank dimensions: a = 1.00, b = 4.00, and c = 1.00.

Aspect Ratio = 1/8

ASPECT RATIO =	Iz/rho		
Theta	20% Full	50% Full	80% Full
0.0	16.859	21.667	26.475
10.0	16.852	21.667	26.481
20.0	16.832	21.667	26.501
30.0	16.792	21.667	26.541
40.0	16.718	21.667	26.616
50.0	16.575	21.667	26.759
60.0	16.259	21.667	27.075
70.0	15.349	21.667	27.984
80.0	12.413	21.667	30.920
90.0	8.7947	21.667	34.539
% Change	48	0	31

Table 6: Iz relative to center of rotation for tank dimensions: a = 1.00, b = 8.00, and c = 1.00.

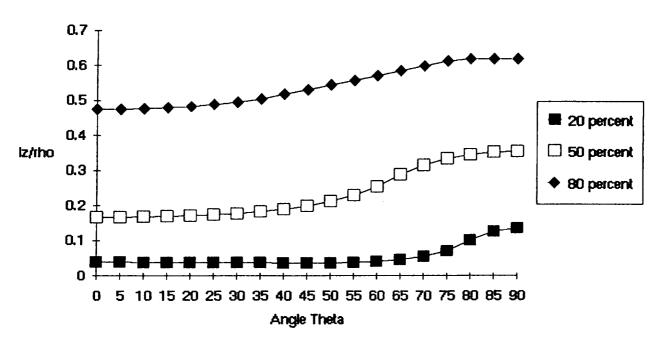


Figure 1: Iz/rho Relative to C.G. of Fluid for 1 \times 2 Tank

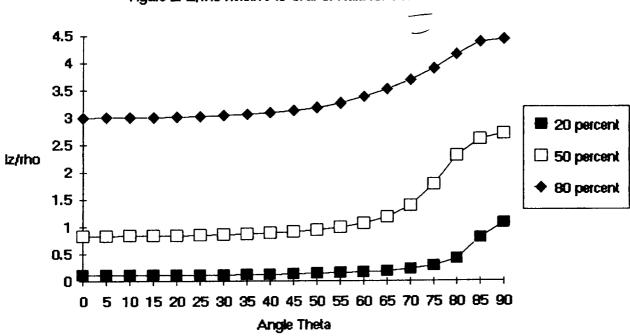


Figure 2: tz/rho Relative to C.G. of Fluid for 1 \times 4 Tank.

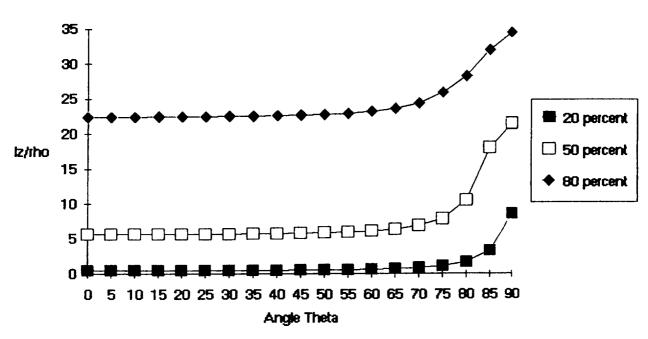


Figure 3: Iz/rho Relative to C.G. of Fluid for 1 \times 8 Tank

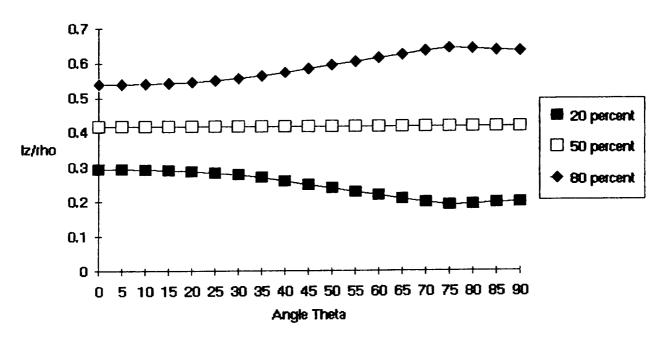


Figure 4: Iz/rho Relative to Center of Rotation for 1x2 Tank

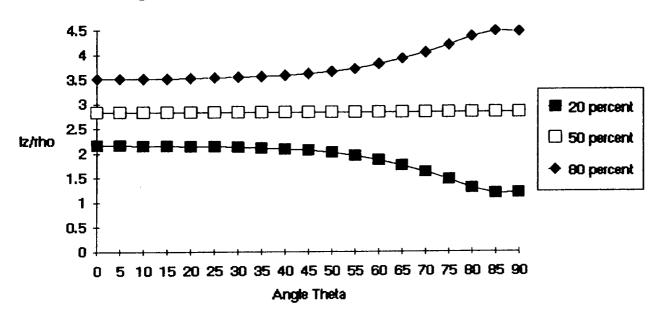


Figure 5: Iz/rho Relative to Center of Rotation for 1 \times 4 Tank

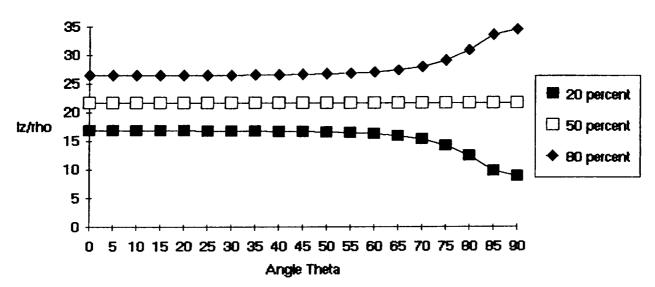


Figure 6: Iz/rho Relative to Center of Rotation for 1 \times 8 Tank

Figure 7: iz/tho of 1 x 2 Tank with Decreasing Fluid

Angle Theta

Figure 8: iz/rho of 1 \times 4 Tank with Decreasing Fluid



Figure 9: tz/rho of 1 \times 8 Tank with Decreasing Fluid

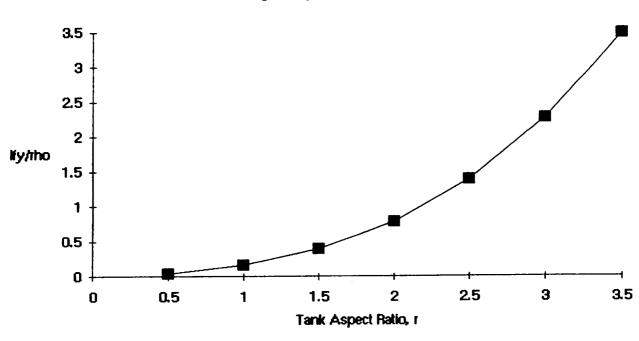
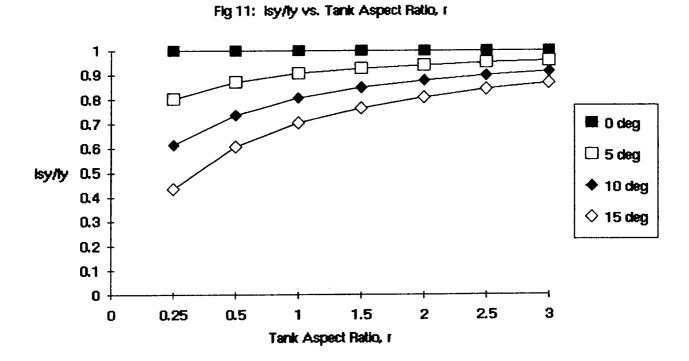


Fig 10: Ify vs. Tank Aspect Ratio, r



Page 1

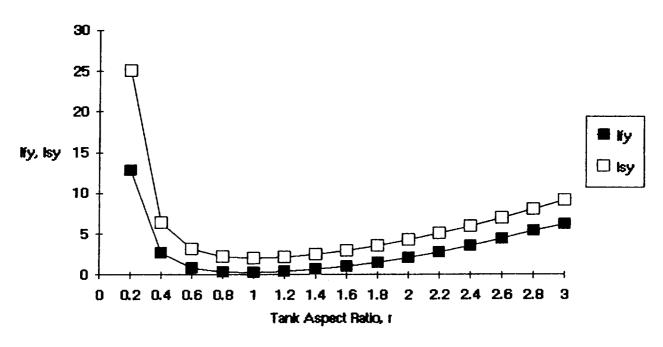


Fig. 13: Ify and Isy vs. Tank Aspect Ratio,r

APPENDIX B PROGRAM LISTINGS

```
ានជាគ្នា ខាណាមកល់អំនៃមកសង់៖
This program calculates the mass moment of inential of the fluid in a tark
relative the to the own, of the fluid. The tank is dimensioned as is war
the x-din. (2) to the y-din(up), and (c) in the z-cirkinto the pages. The
center of the tank is also the center of notation, and corresponds to the
                            The tank notates counterclockwise and this
origin of the coordinates.
abole is defined as theta. This version was last modified on 2/5/90.
   PasPrinter:
onst
   s: = 3.141593:
   the = 1.0:
VD5
   Number = array [1..90] of real:
aг
            :text: (file pointer for output)
   outFile
   outFileName :string[15]:
   output1:Number:
   output2:Number:
   printer:char:
                  (length of tank)
   а.
   b.
                  Cheroht of tank?
                  (width of tank)
   ancie.
                  (the angle of rotation in degrees)
                  (total area of fluid in xx plane)
   area.
                 (Increment by which angle increases)
   increment.
                  (Max area of triangluar shape)
   aTriangle.
   bTr:angle.
                  (Max area of triangle and rectangle combined)
                  (Fluid level as a percent of height "b")
   percent.
                  (Mass of fluid)
   mass.
                 (Mass moment of inertia of entire shape)
   moment.
                 -{2 - theta}
   beta.
   tangentBeta.
                 (tan(beta))
   tangentTheta.
                 (tan(theta))
                  (Angle of rotation CCW in radians)
   theta.
   zero:real:
   form:integer:
                 -{defines the geometry}
nocedure DounterRotate(var ax, ay, ax1, ay1:real);
This procedure converts local Cord, to plobal Cord, given angle theta.)
   ax := (ax1 * cos(theta)) - (ay1 * sin(theta));
   av := (ax1 * sin(theta)) + (av1 * cos(theta));
nd:
inction IBarTri(var mTri,ta,tb:real):real;
This function will calculate the mass moment of inertia of a triangular
orism given its mass, base, and height.
  IBarTri := (mTri/18) * (sgr(ta) + sgr(tb));
inction IBarRect(var mRect,ra,rb:real):real;
This function will calculate the mass moment of inertia of a rectangular
orism given its mass, length, and height.
  IBanRect := (mRect/12) * (sgr(ra) + sgr(rb));
:d:
```

ocedure Centroid(var

xBar,yBar, a1,x1,y1, a2,x2,y2, a3,x3,y3:real);

```
This procedure will calculate the x and y scond of the centroid of the
ficis.
   xArea.
   yAreatreal:
9010
   xArea := (ai * x1) + (a2 * x2) + (a3 * x3);
   yArea := (ai * yi) + (a2 * y2) + (a5 * y8);
   xBar := xArea/area:
   yBan := yAnea/anea:
adı
unction WhatType:integer:
*This function determines what shape the fluid is in. Type 1 = triangle; *)
*type 2 = triangle and rectangle; and type 3 = one triangle and two
                                                                              *)
                                                                              * )
*rectangles.
3.7
   aArea, alpha, bArea :real;
   aTriangle := 0.5 * sqr(a) * tangentTheta;
   aArea := aTriangle + (a * (b + (a * tangentTheta))):
   alpha := arctan(b/a);
   if tangentTheta (> 0.0 then
       bTriangle := abs((0.5 * sqr(b))/tangentTheta);
       bArea := bTrianole + abs((b * (a - (b/tangentTheta))));
   end:
                               (It's a rectangle.)
   if (angle = 0.0) then
       WhatType := 4
   else if (angle = 90.8) then (It's still a rectangle.)
       WhatType := 5
   else if theta <= alpha then
       if (aTriangle ) area) then
       WhatType := 1
       else if (aArea ) area) and (bArea ) area) then
       WhatType := 2
       else WhatType := 3;
   end
   else
   begin
       if (bTriangle ) area) then
       WhatType := 1
       else if (bArea ) area) then
       WhatType := 6
       else WhatType := 7;
   end:
 d:
 acedure TypeOne:
 This procedure will calculate the mass moment of inertia of a triangular
 Г (50).
   a2, b2, h, j, cx, cy:real;
 gin
   h := sqrt((2 * area)/tangentTheta)
                                        * tangentTheta;
   j := sqrt((2 * area)/tangentTheta);
   a2 := (j/3) - (a/2);
   b2 := (h/3) - (b/2):
   CounterRotate(cx, cy, a2, b2);
   moment := IBarTri(mass,h,j);
```

```
This Procedure will taiculate the mass moment of therts of a volume which)
can be broken down into one thiangular, and one rectangular, prism.
   areaTri.
   areaRecti.
               (Height of Rect)
              (Mass of triangular prism)
   massTri.
              (Mass of Rect. prism)
   massRect,
   momentTri, (Moment of inertia of triangular prism rel. to origin)
   momentRect,(Moment of inertia of Rect. prism rel. to origin)
   j, qx, qy.
   X_{\bullet}Y_{\bullet}
   xCen.
   yCen.
   x2, y2,
   x3, y3,
   x4, y4,
   x5, y5:real;
egin
   h := (area - atriangle)/a;
   j := a * tanoentTheta:
   areaTri := atriangle:
   areaRecti := area - aTriangle;
   massTri := areaTri * : * rho;
   massRect := areaRect1 * c * rho;
   qx := a/3 - a/2;
   gy := h + j/3 - b/2:
   CounterRotate(x2,y2,gx,gy);
   x := 0;
   y := b/2 - b/2;
   CounterRotate(x3,y3,x,y);
   Centroid(xCen,yCen,areaTri,x2,y2,areaRect1,x3,y3,zero,zero);
   x4 := x8en - x2;
   y4 := yCen - y2;
   x5 := xCen - x3;
   y5 := yCen - y3:
   momentTri := IBarTri(massTri,a,j) + (massTri * (sgr(x4) + sqr(y4)));
   momentRect := IBarRect(massRect,a.h) + (massRect * (sqr(x5) + sqr(y5)));
   moment := momentTri + momentRect;
nd:
rocedure TypeThree:
This Procedure will calculate the mass moment of inertia of a volume which)
must be broken down into one triangular and two rectangular prisms.
  h. j. K. m.
  areaTri.
  areaRect1.
  areaRect2.
              (mass of triangle)
  massTri.
             (mass of rectangle below triangle)
  massRecti.
              (mass of rectangle next to triangle)
  massRect2.
  momentTri.
  momentRect1.
  momentRect2.
  tri.
               {the triangular area not filled with fluid}
  xCen, yCen, (centroidal coord of the fluid)
  x1. y1.
  x2, y2,
  x3, y3,
  x4, y4,
  x5, y5,
  x6, y6,
  x7, y7,
  х8, у8,
  x9. y9.
  moment1, moment2:real:
```

GROWN FRAGE IS

CH FUCH QUALITY

```
egin
    th: := (a + b) - anea;
    i := sert((2 * tri)/:tangentTheta/:
   h := j * tangentTheta:
   k := a - j:
   m := b - h;
   areaTri := 0.5 * h * j;
   areaRecti := a * m;
   areaRect2 := h * k:
   massTri := areaTri * c * rho:
   massRecti := areaRecti * c * rho;
   massRect2 := areaRect2 * c * rho:
   x2 := k + j/3 - a/2;
   y2 := b/2 - (2 * h/3);
   CounterRotate(x1.y1,x2,y2);
   x5 := 0;
   y5 := m/2 - b/2;
   CounterRotate(x6,y6,x5,y5);
   x4 := K/2 - a/2:
   y4 := 5/2 - h/2:
   CounterRotate(x3,y3.x4,y4):
   Centroid(xCen,yCen,areaTri,x1,y1,areaRect1,x6,y6,areaRect2,>3,y3);
   x7 := xSen - x1:
   y7 := yCen - y1;
   x8 := x0en - xó;
   y8 := yCen − yá;
   x9 := x0en - x3;
   y9 := yCen - y3;
   momentRect1 := IBarRect(massRect1,a,m)+(massRect1 * .sgr(x8) + sgr(y8)));
   momentRect2 := IBarRect(massRect2,h,k)+(massRect2
                                                      * sgr(x9) + sgr(y9)));
   moment := momentTri + momentRect1 + momentRect2;
nd:
rocedure TypeFour;
  c4:real;
eqin
  c4 := percent * 5;
  moment := IBarRect(mass.a.c4):
nocedure TypeFive:
  c5:real;
eain
  c5 := percent * a:
  moment := IBarRect(mass.c5.b);
rocedure TypeSix;
This Procedure will calculate the mass moment of inertia of a volume which)
can be broken down into one triangular and one rectangular prism.
   area2.
                (Area of Rect)
   areaTri.
   areaRecti.
                (Height of Rect)
  massTri.
                (Mass of triangular prism)
  massRect.
                (Mass of Rect. prism)
                (Moment of inertia of triangular prism rel. to origin)
  momentTri.
  momentRect.
                (Moment of inertia of Rect. prism rel. to origin)
   j, gx, gy,
  tri.
               (area of tank not filled with fluid)
```

Х. У.

```
≓Jen, yJen,
    x1, y1.
    жŽ, уŽ,
    х3, у3.
    :4, y4:real:
    areaTri := 0.5 * sqr(b) * tangentBeta;
    areaRect1 := area - areaTri;
    h := (area - areaTri)/b;
    j := b * tangentBeta;
    massTri := areaTri * c * rho;
    massRect := areaRect1 * c * rho:
    gx := h + j/3 - a/2;
   gy := b/3 - b/2;
   CounterRotate(x2,y2,gx,gy);
   x := (h/2) - (a/2):
   у := 0.0;
   CounterRotate(x1.y1.x.y):
   Centroid(xCen.yCen,AreaTri.x2.y2,AreaRect1.x1.y1,zero.zero.zero);
   x3 := x0en - x2:
   y3 := yCen - y2;
   x4 := x0en - x1;
   y4 := y0en - y1;
   momentTri := IBarTri(massTri,b,j) + (massTri * (sqr(x3) + sqr(y3)));
   momentRect := iBarRect(massRect,b,h) + (massRect * (sqr(x4) + sqr(y4)));
   moment := momentTri + momentRect:
rocedure TypeSeven:
This Procedure will calculate the mass moment of inertia of a volume which)
must be broken down into one triangular and two rectangular prisms.
   h. j. K. m.
   areaTri,
   areaRecti.
   areaRect2.
   massTri.
              (mass of triangle)
   massRecti.
              (mass of rectangle below triangle)
              (mass of rectangle next to triangle)
   massRect2.
   momentTri.
   momentRect1.
   momentRect2.
   tri.
               (the triangular area not filled with fluid)
   xCen, yCen.
   axi, ayi,
   ax2, ay2.
   ax3, ay3,
   x1, x2, x3, x4, x5, x6,
   yi, y2, y3, y4, y5, y6,
   moment1, moment2:real;
egin
   tri := (a * b) - area;
   j := sqrt((2 * tri)/tangentBeta);
   h := j * tangentBeta;
   k := b - j;
                                                                              ORIGINAL PACE IS
  m := a - h:
                                                                             OF POOR QUALITY
   areaTri := 0.5 * h * j:
  areaRect1 := b * m:
  areaRect2 := h * k;
  massTri := areaTri * c * rho:
  massRect1 := areaRect1 * c * rho;
  massRect2 := areaRect2 * c * rho;
  \times 2 := a/2 - (2 * h/3);
  y2 := b/2 - (2 * j/3);
  CounterRotate(x1,y1,x2,y2);
  x5 := m/2 - a/2;
  у5 := 0:
```

```
CounterRotate x6,ya.x5,x5);
   34 := 4/2 - 5/2:
   94 := K/2 - 5/2:
   CounterRotate .. 3, y3, x4, y4):
   Centrold(xCen.yCen.areaTri,x1,y1,areaRect1,x5,y6,areaRect2,x3,y3);
   axi := xCen - xi;
   ayi := yCen - yi;
   ax2 := xCen - x6:
   ay2 := yCan - y6;
   ax3 := x0en - x3;
   ay3 := yCen - y3;
   momentTri := IBarTri(massTri,h,j)+(massTri * (sqr(ax1) + sqr(ay1)));
   momentRecti := IBarRect(massRecti.b.m)+(massRecti
                                                    * (sqn(ax2)+sqn(ay2)));
   momentRect2 := IBarRect(massRect2,b,k)+(massRect2
                                                    * (sqr(ax3)+sqr(ay3)));
  moment := momentTri + momentRect1 + momentRect2;
36:
edic (*START MAIN PROGRAM*)
  ClearScreen:
  zero := 0:
  writeln ('Enter dimensions for rectangular tank:'):
  readin (a.t.:);
  writein ('Enter water level as a percentage of "b".'):
  readin (percest);
  while ((percent (= 0)) do
      writeln(10.0 ( Water Level ( 1.0. Try again.1);
      readin(percent);
  end:
  writeln('Enter angle theta increment.');
  readIn(increment):
  writeIn('Enter output file name.');
  readIn(outFileName);
  angle := 0.0;
  area := a * b * percent;
  mass := a * percent * b * d;
  rewrite(outFile,outFileName);
  writeln(outFile,'A = ',a:8:4);
  writelm(outFile./B = /.b:8:4):
  writeln(outFile,'C = ',c:8:4);
  writeln(outFile, 'Percent full = '.percent:B:4);
  writeln(outFile, Type Theta
                                 Moment();
  while (angle (= 90.0) do
  begin
      moment := 0.0;
      beta := ((90 - angle)/360)*(2 * pi);
      theta := (angle/360)*(2 * pi);
      tangentBeta := sin(beta)/cos(beta);
      tangentTheta := sin(theta)/cos(theta);
      form := WhatType;
      case (form) of
          1:TypeOne;
          2:TypeTwo:
          3:TypeThree;
          4:TypeFour;
          5:TypeFive;
          6:TypeSix:
          7:TypeSeven:
      end:
      writeln(outFile,form, ' ',angle:6:2, ' ',moment:8:4);
      angle := angle + increment;
  end:
  close(outFile);
c. (*END MAIN PROGRAM*)
```

The second of the second

```
Program Momenti:
This program calculates the mass moment of inertia of the fluid in a tank
irelative the to the center of the tank. The tank is dimensioned as 'a' in 3
(the x-dir, 'b' in the y-dir(up), and 'c' in the z-dir(into the page). The }
(center of the tank is also the center of rotation and corresponds to the
                            The tank rotates counterclockwise and this
                                                                             1
forigin of the coordinates.
langle is defined as theta. This version was lase modified on 2/4/90.
                                                                             }
Jses
   PasPrinter;
Const
   pi = 3.141593;
   rho = 1.0;
ype
   Number = array [1..90] of real;
lar.
               :text; (file pointer for output)
   outFile
   outFileName :string[15];
   output1:Number;
   output2:Number;
   printer:char;
                   (length of tank)
   a,
   ь,
                   (height of tank)
                   (width of tank)
   τ,
   area,
                   (the angle of rotation in degrees)
   angle,
                   (Increment by which angle increases)
   increment.
                   (Max area of triangluar shape)
   aTriangle,
                   (Max area of triangle and rectangle combined)
   bTriangle,
                  (Fluid level as a percent of height "b")
   percent,
                  (Mass of fluid)
   mass,
                  (Mass moment of inertia of entire shape)
   moment,
                  (2 - theta)
   beta.
                  {tan(beta)}
   tangentBeta,
                  (tan(theta))
   tangentTheta,
                  (Angle of rotation CCW in radians)
   theta:real;
                  (defines the geometry)
   form:integer:
rocedure CounterRotate(var ax, ay, ax1, ay1:real);
This procedure converts local Cord. to global Cord. given angle theta.}
egin
   ax := (ax1 * cos(theta)) - (ay1 * sin(theta));
   ay := (axi * sin(theta)) + (ayi * cos(theta));
nd;
unction IBarTri(var mTri,ta,tb:real):real;
This function will calculate the mass moment of inertia of a triangular
prism given its mass, base, and height.
egin
   18arTri := (mTri/18) * (sqr(ta) + sqr(tb));
nd;
unction IBarRect(var mRect,ra,rb:real):real;
This function will calculate the mass moment of inertia of a rectangular
                                                                             }
prism given its mass, length, and height.
   IBarRect := (mRect/12) * (sqr(ra) + sqr(rb));
10:
```

+This function determines what shape the fluid is in. Type 1 = triangle;

ttype 2 = triangle and rectangle; and type 3 = one triangle and two

inction WhatType:integer;

ORIGINAL PAGE IS OF POOR QUALITY

*)

*)

¥١

```
Jar
   aArea, alpha, bArea :real;
egin
    aTriangle := 0.5 * sqr(a) * tangentTheta;
   aArea := aTriangle + (a * (b - (a * tangentTheta)));
   alpha := arctan(b/a);
   if tangestTheta <> 0.0 then
   begin
        bTriangle := abs((0.5 * sqr(b))/tangentTheta);
        bArea := bTriangle + abs((b * (a - (b/tangentTheta))));
   end;
   if (angle = 0.0) then
                                 (It's a rectangle.)
       WhatType := 4
   else if (angle = 90.0) then (It's still a rectangle.)
       WhatType := 5
   else if theta <= alpha then
   begin
        if (aTriangle ) area) then
       WhatType := 1
        else if (aArea ) area) and (bArea ) area) then
       WhatType := 2
        else WhatType := 3;
   end
   eise
   begin
       if (bTriangle ) area) then
       WhatType := 1
       else if (bArea ) area) then
       WhatType := 6
       else WhatType := 7;
   end;
nd;
rocedure TypeOne;
This procedure will calculate the mass moment of inertia of a triangular
prism.
ar
   a2, b2, h, j, cx, cy:real;
egin
   h := sqrt((2 * area)/tangentTheta)
                                        * tangentTheta;
   j := sqrt((2 * area)/tangentTheta);
   a2 := (j/3) - (a/2);
   b2 := (h/3) - (b/2);
   CounterRotate(cx, cy, a2, b2);
   moment := IBarTri(mass,h,j) + (mass *(sqr(cx) + sqr(cy)));
nd:
rocedure TypeTwo:
This Procedure will calculate the mass moment of inertia of a volume which)
can be broken down into one triangular and one rectangular prism.
aг
               (Area of Rect)
   area2,
               (Height of Rect)
   massTri.
               (Mass of triangular prism)
   massRect.
              (Mass of Rect. prism)
   momentTri, {Moment of inertia of triangular prism rel. to origin}
   momentRect, (Moment of inertia of Rect. prism rel. to origin)
   j, gx, gy, xbar, ybar,
   x, y, x3, y3:real;
egin
   h := (area - atriangle)/a;
   j := a * tangentTheta;
   massTri := atriangle * c * rho;
   massRect := (area - atriangle) * c * rho;
   gx := a/3 - a/2;
   ny := h + i/3 - b/2:
```

```
CounterRotate(xbar,ybar,gx,gy);
   momentTri := IBarTri(massTri,a,j) + (massTri * (sqr(xbar) + sqr(ybar)));
    x := 0;
    y := h/2 - b/2;
    CounterRotate(x3,y3,x,y);
    momentRect := IBarRect(massRect,a,h) + (massRect * (sqr(x3) + sqr(y3)));
   moment := momentTri + momentRect;
and:
Procedure TypeThree;
iThis Procedure will calculate the mass moment of inertia of a volume which)
imust be broken down into one triangular and two rectangular prisms.
    h, j, k, m,
                (mass of triangle)
    massTri,
                (mass of rectangle below triangle)
    massRect1,
               {mass of rectangle next to triangle}
    massRect2,
   momentTri.
   momentRect1,
   momentRect2,
                (the triangular area not filled with fluid)
    tri,
    x1, x2, x3, x4, x5, x6,
    y1, y2, y3, y4, y5, y6,
    moment1, moment2:real;
pegin
    tri := (a * b) - area;
    j := sqrt((2 * tri)/tangentTheta);
    h := j * tangentTheta;
    k := a - j;
    m := b - h;
   massTri := 0.5 * h * j * c * rho;
   massRect1 := a * m * c * rho;
   massRect2 := h * k * c * rho;
   x2 := k + j/3 - a/2;
   y2 := b/2 - (2 * h/3);
   CounterRotate(x1,y1,x2,y2);
   momentTri := IBarTri(massTri,h,j) + (massTri * (sqr(x1) + sqr(y1)));
   x5 := 0;
    y5 := m/2 - b/2;
    CounterRotate(x6,y6,x5,y5);
                                                         * (sqr(x6) + sqr(y6)));
    momentRect1 := IBarRect(massRect1,a,m)+(massRect1
    x4 := k/2 - a/2;
    y4 := b/2 - h/2;
    CounterRotate(x3,y3,x4,y4);
                                                         * (sqr(x3) + sqr(y3)));
    momentRect2 := IBarRect(massRect2,h,k)+(massRect2
    moment := momentTri + momentRect1 + momentRect2;
ind:
Procedure TypeFour;
Har
    c4:real;
egin
    c4 := percent * b;
    moment := IBarRect(mass,a,c4) + (mass * sqr((b/2) * (percent - 1)));
nd:
'rocedure TypeFive;
   c5:real;
egin
    c5 := percent * a;
    moment := IBarRect(mass,c5,b) + (mass * (sqr((a/2) * (percent - 1))));
```

nd;

```
Procedure TypeSix;
 (This Procedure will calculate the mass moment of inertia of a volume which)
 {can be broken down into one triangular and one rectangular prism.
 var
                  (Area of Rect)
     area2.
     areaTriangle,
                  (Height of Rect)
                  (Mass of triangular prism)
     massTri,
                  (Mass of Rect. prism)
     massRect,
     momentTri.
                  (Moment of inertia of triangular prism rel. to origin)
     momentRect,
                  (Moment of inertia of Rect. prism rel. to origin)
     j, gx, gy, xbar, ybar,
     x, y, x3, y3:real;
 begin
     areaTriangle := 0.5 * sqr(b) * tangentBeta;
     h := (area - areaTriangle)/b;
     j := b * tangentBeta;
    massTri := areaTriangle * c * rho;
    massRect := (area - areaTriangle) * c * rho;
    gx := h + j/3 - a/2;
    gy := b/3 - b/2;
    CounterRotate(xbar,ybar,gx,gy);
    momentTri := IBarTri(massTri,b,j) + (massTri * (sqr(xbar) + sqr(ybar)));
    x := (h/2) - (a/2);
    y := 0.0;
    CounterRotate(x3,y3,x,y);
                                           + (massRect * (sqr(x3) + sqr(y3)));
    momentRect := iBarRect(massRect,b,h)
    moment := momentTri + momentRect;
end:
Procedure TypeSeven;
(This Procedure will calculate the mass moment of inertia of a volume which)
(must be broken down into one triangular and two rectangular prisms.
var
    h, j, k, m,
                 (mass of triangle)
    massTri,
                 (mass of rectangle
                                    below triangle)
    massRect1,
    massRect2,
                 (mass of rectangle next to triangle)
    momentTri,
    momentRecti,
    momentRect2,
                (the triangular area not filled with fluid)
    x1, x2, x3, x4, x5, x6,
    y1, y2, y3, y4, y5, y6,
    moment1, moment2:real;
nigec
    tri := (a * b) - area;
    j := sqrt((2 * tri)/tangentBeta);
    h := j * tangentBeta;
    k := b - j;
    m := a - h;
   massTri := 0.5 * h * j * c * rho;
    massRecti := b * m * c * rho;
   massRect2 := h * k * c * rho;
   x2 := a/2 - (2 * h/3);
   y2 := b/2 - (2 * j/3);
   CounterRotate(x1,y1,x2,y2);
                                                                                           ORIGINAL PAGE IS
   momentTri := IBarTri(massTri,h,j)+(massTri * (sqr(x1) + sqr(y1)));
                                                                                          OF POOR QUALITY
   x5 := m/2 - a/2;
   y5 := 0;
   CounterRotate(x6,y6,x5,y5);
   momentRect1 := IBarRect(massRect1,b,m)+(massRect1
                                                      * (sqr(x6) + sqr(y6)));
   x4 := a/2 - h/2;
   y4 := k/2 - b/2;
   CounterRotate(x3,y3,x4,y4);
   momentRect2 := IBarRect(massRect2,h,k)+(massRect2
                                                        * (sqr(x3) + sqr(y3)));
   moment := momentTri + momentRect1 + momentRect2;
```

```
Begin (*START MAIN PROGRAM*)
     ClearScreen:
    writeln ('Enter dimensions for rectangular tank:');
    readln (a,b,c);
    writeln ('Enter water level as a percentage of "b".');
    readln (percent);
    while ((percent )= 1) or (percent (= 0)) do
    begin
         writeln('0.0 < Water Level < 1.0. Try again.');
        readin(percent);
    end:
    writeln('Enter
                    angle theta increment.');
    readln(increment);
    writeln('Enter
                   output file name.');
    readln(outFileName);
    angle := 0.0;
    area := a * b * percent;
    mass := a * percent * b * c;
    rewrite(outFile,outFileName);
    writeln(outFile,'A = ',a:8:4);
writeln(outFile,'B = ',b:8:4);
writeln(outFile,'C = ',c:8:4);
    writeln(outFile,'Percent full = ',percent:8:4);
    writeln(outFile,'Type
                             Theta Moment');
    while (angle <= 90.0) do
    begin
        moment := 0.0;
        beta := ((90 - angle)/360)*(2 * pi);
        theta := (angle/360)*(2 * pi);
        tangentBeta := sin(beta)/cos(beta);
        tangentTheta := sin(theta)/cos(theta);
        form := WhatType;
        case (form) of
            1:TypeOne;
            2:TypeTwo;
            3:TypeThree;
            4:TypeFour;
            5:TypeFive;
            6:TypeSix;
            7:TypeSeven;
        end;
                                   ',angle:6:2, ' ',moment:8:4);
        writeln(outFile,form, '
        angle := angle + increment;
   end;
   close(outFile);
End. (*END MAIN PROGRAM*)
```

APPENDIX B
PROGRAM LISTINGS

```
nooram infomentüflmentia:
Firs program calculates, the mass moment of inential of the fluid in a tank relative, the to the c.p. of the fillog. The tank is dymensioned, as is [p]
the redin. I so the sederium , and for a the decipients the cases. The
cepter of the tark is also the depter of notation and connesponds to the origin of the coordinates. The tank notates countend ockwise and this
onique of the coordinates. The tank notates counterclockwise and this store is defined as theta. This wers on was last modified on 2/5/90.
  -PasPriater:
  | p: | = | 3.040593:
  rho = 1.0;
  Number = array [1..90] of real:
a:
   outfile itexts (file pointer for output)
   outFileName :string1.5):
   outoutl:Number:
   output2:Number:
   on interichan;
                    (length of tank)
                    (height of tank)
   ъ.
                    (width of tank)
                   (the anole of notation in decrees)
   ancie.
                   litotal area of fiuld in my plane.
   increment.
                    (Granement by which anole increases)
   aThiangle,
                   KMax area of thrangilar shapel
   ວີກ ເສດຊະຊາ
                   (Max area of thishgle and rectangle combined)
                   (Fluxo level as a percent of height "5")
   dencent.
                   (Mass of fibid)
   files.
                   -{Mass owners of Epentia of entire shape:
   moment.
                   (2 - theta)
   beta.
   tangentBeta,
                   (tan(beta))
   tangentTheta.
                    (tan(theta))
                    (Angle of notation CCW in madians)
   toeta.
   zencinea'i
   form:integer:
                 idefines the geometry?
hodadura CounterRotate(van ax, ax, axi, ayi:neal);
This procedure converts local Cord. to plobal Cord. given angle theta.)
  ax := (axi * cos(theta)) - (ayi * sin(theta));
  av := (ax1 * sin(theta)) + (ay1 * cos(theta));
nd;
unction [BarTri(var mTri,ta.tb:real):real;
This function will calculate the mass moment of inertia of a triangular
prism given its mass, base, and height.
ecin
 IBarTri := (mTri/18) * (sgr(ta) + sgr(tb));
nd:
unction IBarRect(var mRect.ra.rb:real):real:
This function will calculate the mass moment of inertia of a rectangular
orism given its mass, length, and height.
 nd:
```

nocedure Centroid(var xBar,yBar,

a1,x1,y1. a2,x2,y2, a3,x3,y3:real);

```
This procedure will calculate the x and x coord of the centrold of the
fluis.
   KArea.
   yAneainea}:
   NAmea (# (a1 * x1) + (a2 * x2) + (a3 + x3);
   yApea := (a( * v1) + (a2 * v2) + (a3 * v5::
   xBar := xArea/area;
   yBan := yAnea/anea:
and:
Function WhatType:integer:
*This function determines what shape the fluid is in. Type 1=triangle;\;\;*)
*type 2 = triangle and rectangle: and type 3 = one triangle and two
                                                                              *)
*rectangles.
                                                                              * 1
   aAnea, alpha, bAnea :real;
   aTriangle := 0.5 * sgr(a) * tangentTheta;
   aAnea := aTh:anole + /a * (b - (a * tanoentTheta))):
   aloha := anctan(b/a);
    if tangentTheta <0 0.8 then</pre>
    28510
       cPhlangle := abs00.5 * sqrvppb/tangentThetable
        bAnea := bTrvansle + abs((b + va - ib/tancentTheta )));
     f (angle = 0.0) then
                             (It's a nectangle.)
       whatType := 4
    else if vangle = 90.0) then (It's still a restancie...
       WhatType := 5
   alse if theta (= alpha then
        of (aThiangle - 2 area), then
       WhatType := 1
        else of caArea ) area) and (sArea / area) then
       whatTybe := 2
        else WhatType := 3;
   end
   e 13e
   beain
        if (biriangle > area) then
       WhatType := 1
        else if (bArea ) area) then
       WhatType := 6
        else WhatTupe := 7;
   end:
ind:
Procedure TypeOne:
This procedure will calculate the mass moment of inertia of a triangular
                                                                               }
.prism.
   a2, b2, h, j, cx, cy:real;
eoin
   h := sort((2 * area)/tangentTheta; * tangentTheta;
                                                                                  ORIGINAL PAGE IS
   j := sqrt((2 * area)/rangentTheta);
   a2 := (3/3) - (a/2);
                                                                                  OF POOR QUALITY
   b2 := (h/3) - (b/2);
   CounterRotate(cx, cy, a2, b2);
   moment := IBarTri(mass,h,j):
nd;
```

rocedure TypeTwo:

```
This Procedure will calculate the mass moment of inertial of a volume which:
can be broken down into one thrangular and one rectangular prism.
   aneaThi.
   areaRecti.
               (Height of Rect)
              (Mass of thiangular prism)
              (Mass of Rect. prism)
   massRect.
   momentTri, (Moment of inertia of triangular prism rel. to origin)
   momentRest. (Moment of inertia of Rest. prism rel. to origin)
   j, gx, gy.
   x_*y_*
   жСел.
   yCen,
   x2, y2,
   ж3, у3,
   x4, y4,
   x5. y5:real:
egin
  |h|:= (area - atriangle)/a;
   j := a * tangentTheta:
   areaTr: := atriangle:
   areaRecti := area - aTriangle:
massTri := areaTri * : * rho:
   masshedt (# aneshadt) 🕭 d 🛎 cho:
   3x := a/3 - a/2:
   gx := t - [/S - b/2:
   |Causis=Fotate(x2.-2.gx,py):
   # != €:
   s := 5.72 - 5.2;
   CounterRotate(x3.98.8.9):
   |Senthold(xSen,ySen,areaTri,x2,y2.areaRecti,x3,y3.teno.teno,zeno);
   34 := xCen - x2;
   y4 := yCen - >1:
   %5 ;= %Cen - £8;
   √5 := √2en - √3;
   momentTri := iBarTri(massTri,a.j) + (massTri * (sgr.x4) + scr(x4)));
   momentRept := IBanReptimassRept.a.h) - (massRept * (son(x5) + son(x5))):
   moment := momentTri + momentRect;
361
nocedure TypeThree:
This Procedure will calculate the mass moment of inertia of a volume which)
must be broken down into one triangular, and two rectangular, prisms,
   Դ. J. X. ಮ.
   areaTri.
   areaRecti.
   areaRect2.
  massTri,
               (mass of triangle)
  massRecti,
              (mass of rectangle below triangle)
  massRect2, (mass of rectangle next to triangle)
  momentīri.
  momentRect1.
  momentRect2.
   tri.
               (the triangular area not filled with fluid)
   xien. yien. (centroidal coord of the fluid)
   x1, y1,
   82, y2,
   x3, y3,
   x4, y4,
   x5, y5,
   x6, y6,
   x7, y7,
   ж8, у8.
   x9, y9,
```

momenti, moment2:real;

```
edin
   thi := (a * b) - anea;
   j := sort((2 * tri)/tangentTheta/;
   h := j * tangentTheta;
   1 1= 2 - 1
   m := 5 - h:
   areaTri := 0.5 + h + j:
   areaRectl := a * m;
   areaRest2 := h * K:
   massTr! := areaTr! * c * rho;
   massRecti := areaRecti * c * rhc:
   massRect2 := areaRect2 * c * rho:
   x2 := k + j/3 - a/2;

y2 := b/2 - (2 * b/3);
   CounterRotate(x1.v1,x2.v2):
   x5 := 0:
   y5 := m/2 - 5/2;
   CounterRotate(x6,y6,x5,y5):
   x4 := x/2 - x/2:
   y4 := 5/2 - 5/2:
   CounterRotate(x3,y3.x4,y4):
   Centroid(xCen,yCen,areaTri,x1,y1,areaRect1,x6,y6,areaRect2,x3,y3);
   x7 := xlen - x1:
   y7 := \Jer - y1:
   %8 := x8et - x¢;
   ∀2 := ySen - √6:
   49 i= 45er - 55t
   v≎ := /2er - 3:
   momencine := iBanine(tassine, to, the constant <math>+ esch(nil) + sqn(vil));
   ★ kegn(x9) + edfty9):::
  | momentRect2| | := | IBanRect(massRect2.f.,E:+kmassFest2|
  - moment := momentTri  + momentRect1  + momentRect2:
hosedure TypeFour:
  [64::eal;
edin
  | c4 := percent * b;
  moment := IBanRect(mass.a.c4):
nd:
nocedure TypeFive:
  c5:real;
  | c5 := percent * a:
  moment := IBarRect(mass,c5,b):
nd:
nocedure TypeSix:
This Procedure will calculate the mass moment of inertia of a volume which)
can be broken down into one triangular and one rectangular prism.
                (Area of Rest)
   areaZ.
   aneaTri.
   areaRecti.
                (Height of Rect)
                (Mass of triangular prism)
   massTri.
   massRect.
                (Mass of Rect. prism)
   momentTri.
                (Moment of inertia of triangular prism rel. to origin)
   momentRect.
                (Moment of inertia of Rect. prism rel. to origin)
   j, gx, gy,
   tri.
                (area of tank not filled with fluid)
```

Х. У.

```
xlen, yûen,
   His Fig.
   a2. v2.
   ,S. vI.
    14. y4:real:
   areaTr := 0.5 % son.b/ % tangentBeta;
   areaRect) := area - shea?cr:
   h := (anea - aneaThi)/b;
   ; := b * tangentBeta;
   massTri := areaTri * c * rho:
   massRect := areaRect1 * c * rho:
   ax = b + 1/3 - a/2:
   qy := 5/3 - 5/2:
   CounterRotate(x2,x2,gx,gy);
   x := xb/2x - xa/2x:
   y := 0.0;
   JounterRotate(x1,y1,x.v.;
   Centrold(xJen, yCen, AreaTr), x2, x2, AreaRecti, x1, y1, zero, zero);
   3.3 := xCat - x2:
   y3 := y0en - y2;
   x4 (= xCen - x1)
   ∀4 := y0er - y1:
   momentTr: := IBanTr:(massTri,b,j) + (massTri * (sqr(x3) + sqr(y3)));
   momertRept := {BarRestymassRept.b.b} + (massRept * (sqr(y4) + sqr(y4));
  moment := memertTr:  + momentReat:
nacecure TypeSeven;
The Procedure will
                    calculate the mass moment of inertia of a volume which?
must be proxed bown into one thishquian and two rectangular prisms,
  . B., . , K., B.,
  areaTr ..
   aneaRecti,
   areaRectl.
   massīni,
               (mass of thrangle)
   massResti.
              Table of rectangle below triangle)
   massRect2,
              (mass of rectangle next to triangle)
   momentTra.
   momentRectl.
   momentRect2.
   t-1.
                (the triangular area not filled with fluid)
   xCen. yCen.
   ax1, ay1,
   ax2, ay2.
   ax3, ay3,
   x1, x2, x3, x4, x5, x6,
   yi, y2, y3, y4, y5, y6,
   moment1, moment2:real;
egin
   tri := (a * b) - area;
   j := sqrt((2 * tri)/tangentBeta):
   ት := j * tangentBeta:
   k := b - j;
   m := a - h:
   areaTri := 0.5 * h * j:
  areaRect1 := b * m;
  areaRect2 := h * h:
  massTri := areaTri * c * rho;
  massRecti := areaRecti * c * rho:
  massRect2 := areaRect2 * c * rno:
  3.2 := a/2 - (2 * h/3):
  y2 := 5/2 - (2 * j/3);
  CounterRotate(x1,y1.x2,y2):
  x5 := \pi \sqrt{2} - a/2;
  y5 := 0:
```

```
|CounterRotate(x5.y5.x5.x5.x5);
  34 := 4/2 - 5/2:
  v4 := K/2 - 5/2:
   CounterRotate / 5.v3.04.v4%:
   CentrondixGen, wDen, areaTri, x1, y1, areaRect1, v5, v6, areaRect2, x8, y8);
  axi /= xCer - xi:
  ayi := y040 - 511
  ax2 := xJen - xe:
  ayî := /Can - ya:
  ax8 := +len - x8:
   ay3 := y0en - y3;
  momentTri := IBarTri(massTri,h,j)+(massTri + (sqr(ax1) + sqr(ay1)));
  momentRect2 := IBanRect(massRect2.b,k)+(massRect2.b)
                                                   * (sqr(ax3)+sqr(ay3)));
  moment := momertTri + momertRect1 + momentRect2:
eoir (*START MAIN PROSPAM*)
   ClearScheen;
   zeno := 0:
  writein ('Enter dimensions for restangular tank:'):
  readin (a.t.t):
  whitele Enter water Tevel as a percentage of "b". 12;
  neadin (pencent):
  while ((percent 0= 1) or (percent (= 8)) do
  262:5
      whiteln("5.1 - C Water Level | 0.1.0. Thy again."):
      read mimencent:
  whiteleffExten angle theta increment. A:
  neadln(increment):
  whitein((Enter | output file name.();
  readin(outFileName):
  angle := 0.0:
  area := a + b + pencent:
  mass := a * percent * b # c:
  rewrite(outFile.outFileName):
  writein(outFile,'A = ',a:\hat{\epsilon}:4):
  whitein(outFile./B = \%.b:8:4):
  whiteln(outFile,'C = ',c:8:4);
  writein(outF)le, 'Fercent full = '.percent:8:4);
                        Theta Moment();
  writeln(outFile,/Type
  while (anole (= 90.0) do
  begin
      moment := 0.0;
      beta := ((90 - angle)/360)*(2 * p!);
      theta := (angle/360)*(2 * pi);
      tangentBeta := sin(beta)/cos(beta);
      tangentTheta := sin(theta)/cos(theta);
      form := WhatType;
      case (form) of
          1:TypeOne:
          2:TypeTwo:
          3:TypeThree;
          4:TypeFour:
          5:TypeFive;
                                                                          ORIGINAL PAGE IS
          δ:TypeSix:
                                                                          OF POOR QUALITY
          7:TypeSeven;
      end:
      writeln(outFile,form, / /,angle:6:2, / /,moment:8:4);
      angle := angle + increment;
  and:
  close(outFile);
nd. (*BND MAIN PROGRAM*)
```

```
Program Momentl:
This program calculates the mass moment of inertia of the fluid in a tank
relative the to the center of the tank. The tank is dimensioned as (a) in 3
(the x-dir, 'b' in the y-dir(up), and 'c' in the z-dir(into the page). The )
center of the tank is also the center of rotation and corresponds to the
lorigin of the coordinates. The tank rotates counterclockwise and this
langle is defined as theta. This version was lase modified on 2/4/90.
    PasPrinter:
Const
    pi = 3.141593;
    rho = 1.0;
Type
    Number = array [1..90] of real:
Jan
    outfile
               :text; {file pointer for output}
    outFileName :string[15];
    output1:Number;
    output2:Number;
    printer:char;
                   (lenoth of tank)
    a,
                   (height of tank)
    ь,
    c,
                   (width of tank)
    area,
    angle,
                   (the angle of rotation in degrees)
                   (Increment by which angle increases)
    increment,
                   (Max area of triangluar shape)
    aTriangle.
    bTriangle,
                   (Max area of triangle and rectangle combined)
                   (Fluid level as a percent of height "b")
    percent,
                   (Mass of fluid)
   mass,
                   (Mass moment of inertia of entire shape)
    moment.
   beta,
                   \{2 - \text{theta}\}
    tangentBeta,
                   (tan(beta))
                   (tan(theta))
    tangentTheta,
                   (Angle of rotation CCW in radians)
    theta:real;
    form:integer;
                   (defines the geometry)
Procedure CounterRotate(var ax, ay, ax1, ay1:real);
(This procedure converts local Cord. to global Cord. given angle theta.)
nigec
    ax := (axi * cos(theta)) - (ayi * sin(theta));
    ay := (ax1 * sin(theta)) + (ay1 * cos(theta));
≥nd:
Function IBarTri(var mTri,ta,tb:real):real;
(This function will calculate the mass moment of inertia of a triangular
iprism given its mass, base, and height.
   IBarTri := (mTri/18) * (sqr(ta) + sqr(tb));
end:
-unction IBarRect(var mRect,ra,rb:real):real;
This function will calculate the mass moment of inertia of a rectangular
                                                                             )
                                                                             }
prism given its mass, length, and height.
   IBarRect := (mRect/12) * (sqr(ra) + sqr(rb));
∙nd:
unction WhatType:integer:
*This function determines what shape the fluid is in. Type 1 = triangle;
                                                                            *)
*type 2 = triangle and rectangle; and type 3 = one triangle and two
                                                                            *>
```

*)

¥nortannlee

```
aArea, alpha, bArea :real;
egin
   aTriangle := 0.5 * sqr(a) * tangentTheta;
   aArea := aTriangle + (a * (b - (a * tangentTheta)));
   alpha := arctan(b/a);
   if tangentTheta <> 0.0 then
        bTriangle := abs((0.5 * sqr(b))/tangentTheta);
        bArea := bTriangle + abs((b * (a - (b/tangentTheta))));
   end;
   if (angle = 0.0) then
                                 (It's a rectangle.)
       WhatType := 4
   else if (angle = 90.0) then (It's still a rectangle.)
       WhatType := 5
   else if theta <= alpha then
   begin
        if (aTriangle ) area) then
       WhatType := 1
        else if (aArea ) area) and (bArea ) area) then
       WhatType := 2
        else WhatType := 3;
   end
   else
   begin
        if (bTriangle ) area) then
       WhatType := 1
        else if (bArea ) area) then
       WhatType := 6
        else WhatType := 7;
   end:
∍nd;
Procedure TypeOne;
This procedure will calculate the mass moment of inertia of a triangular
iprism.
ar
   a2, b2, h, j, cx, cy:real;
eqin:
                                        * tangentTheta;
   h := sqrt((2 * area)/tangentTheta)
   j := sqrt((2 * area)/tangentTheta);
   a2 := (j/3) - (a/2);
   b2 := (h/3) - (b/2);
                     cy, a2, b2);
   CounterRotate(cx,
   moment := IBarTri(mass,h,j) + (mass *(sqr(cx) + sqr(cy)));
and:
rocedure TypeTwo;
This Procedure will calculate the mass moment of inertia of a volume which)
can be broken down into one triangular and one rectangular prism.
ar.
               (Area of Rect)
   area2,
               (Height of Rect)
               (Mass of triangular prism)
   massTri,
               (Mass of Rect. prism)
   momentTri, (Moment of inertia of triangular prism rel. to origin)
   momentRect, (Moment of inertia of Rect. prism rel. to origin)
   j, gx, gy, xbar, ybar,
   x, y, x3, y3:real;
regin
   h := (area - atriangle)/a;
   j := a * tangentTheta;
   massTri := atriangle * c * rho;
   massRect := (area - atriangle) * c * rho;
   qx := a/3 - a/2;
   qy := h + j/3 - b/2;
```

```
CounterRotate(xbar,ybar,gx,gy);
    momentTri := IBarTri(massTri,a,j) + (massTri * (sqr(xbar) + sqr(ybar)));
    x := 0;
    y := h/2 - b/2;
    CounterRotate(x3,y3,x,y);
                                           + (massRect * (sqr(x3) + sqr(y3)));
    momentRect := IBarRect(massRect,a,h)
    moment := momentTri + momentRect;
and;
Procedure TypeThree;
(This Procedure will calculate the mass moment of inertia of a volume which)
(must be broken down into one triangular and two rectangular prisms. )
    h, j, k, m,
                 (mass of triangle)
    massTri,
              {mass of rectangle below triangle}
    massRect1,
    massRect2, (mass of rectangle next to triangle)
    momentTri,
    momentRect1,
    momentRect2,
                 (the triangular area not filled with fluid)
    tri,
    x1, x2, x3, x4, x5, x6,
    y1, y2, y3, y4, y5, y6,
    moment1, moment2:real;
begin
    tri := (a * b) - area;
    j := sqrt((2 * tri)/tangentTheta);
    h := j * tangentTheta;
    k := a - j;
    m := b - h;
    massTri := 0.5 * h * j * c * rho;
    massRect1 := a * m * c * rho;
    massRect2 := h * k * c * rho;
    x2 := k + j/3 - a/2;
    y2 := b/2 - (2 * h/3);
    CounterRotate(x1,y1,x2,y2);
    momentTri := IBarTri(massTri,h,j) + (massTri * (sqr(x1) + sqr(y1)));
    x5 := 0;
    y5 := m/2 - b/2;
    CounterRotate(x6,y6,x5,y5);
                                                        * (sqr(x6) + sqr(y6)));
    momentRect1 := IBarRect(massRect1,a,m)+(massRect1
    x4 := k/2 - a/2;
    y4 := b/2 - h/2;
    CounterRotate(x3,y3,x4,y4);
                                                         * (sqr(x3) + sqr(y3)));
    momentRect2 := IBarRect(massRect2,h,k)+(massRect2
    moment := momentTri + momentRect1 + momentRect2;
end;
Procedure TypeFour;
var
    c4:real;
    c4 := percent * b;
    moment := IBarRect(mass,a,c4) + (mass * sqr((b/2) * (percent - 1)));
end:
Procedure TypeFive;
var
    c5:real;
begin
    c5 := percent * a;
    moment := 1BarRect(mass,c5,b) + (mass * (sqr((a/2) * (percent - 1))));
```

```
Procedure TypeSix;
(This Procedure will calculate the mass moment of inertia of a volume which)
(can be broken down into one triangular and one rectangular prism.
                 (Area of Rect)
    area2,
    areaTriangle,
                 (Height of Rect)
   h,
   massTri,
                 (Mass of triangular prism)
    massRect,
                (Mass of Rect. prism)
                (Moment of inertia of triangular prism rel. to origin)
    momentTri.
                (Moment of inertia of Rect. prism rel. to origin)
    momentRect,
    j, gx, gy, xbar, ybar,
    x, y, x3, y3:real;
begin
    areaTriangle := 0.5 * sqr(b) * tangentBeta;
    h := (area - areaTriangle)/b;
    j := b * tangentBeta;
   massTri := areaTriangle * c * rho;
   massRect := (area - areaTriangle) * c * rho;
    gx := h + j/3 - a/2;
    gy := b/3 - b/2;
    CounterRotate(xbar,ybar,gx,gy);
   momentTri := IBarTri(massTri,b,j) + (massTri * (sqr(xbar) + sqr(ybar)));
   x := (h/2) - (a/2);
    y := 0.0;
    CounterRotate(x3,y3,x,y);
                                          + (massRect * (sqr(x3) + sqr(y3)));
   momentRect := iBarRect(massRect,b,h)
   moment := momentTri + momentRect;
Procedure TypeSeven;
(This Procedure will calculate the mass moment of inertia of a volume which)
(must be broken down into one triangular and two rectangular prisms.
   h, j, k, m,
                (mass of triangle)
   massTri,
   massRect1,
              (mass of rectangle below triangle)
   massRect2, (mass of rectangle next to triangle)
   momentTri.
   momentRecti,
   momentRect2,
                (the triangular area not filled with fluid)
   x1, x2, x3, x4, x5, x6,
   y1, y2, y3, y4, y5, y6,
   moment1, moment2:real;
begin
    tri := (a * b) - area;
    j := sqrt((2 * tri)/tangentBeta);
   h := j * tangentBeta;
   k := b - j;
   m := a - h;
   massTri := 0.5 * h * j * c * rho;
   massRecti := b * m * c * rho;
   massRect2 := h * k * c * rho;
   x2 := a/2 - (2 * h/3);
   y2 := b/2 - (2 * j/3);
   CounterRotate(x1,y1,x2,y2);
   momentTri := IBarTri(massTri,h,j)+(massTri * (sqr(x1) + sqr(y1)));
   x5 := m/2 - a/2;
   y5 := 0;
   CounterRotate(x6,y6,x5,y5);
   momentRect1 := IBarRect(massRect1,b,m)+(massRect1
                                                      * (sqr(x6) + sqr(y6)));
   x4 := a/2 - h/2;
   y4 := k/2 - b/2;
   CounterRotate(x3,y3,x4,y4);
   momentRect2 := IBarRect(massRect2,h,k)+(massRect2
                                                        * (sqr(x3) + sqr(y3)));
   moment := momentTri + momentRect1 + momentRect2;
```

```
Begin (*START MAIN PROGRAM*)
    ClearScreen;
    writeln ('Enter dimensions for rectangular tank:');
    readin (a,b,c);
    writeln ('Enter water level as a percentage of "b".');
    readin (percent);
    while ((percent )= 1) or (percent <= 0)) do
    begin
        writeln('0.0 ( Water Level ( 1.0. Try again.');
        readln(percent);
    end;
    writeln('Enter
                   angle theta increment.');
    readln(increment);
                   output file name.');
    writeln('Enter
    readln(outFileName);
    angle := 0.0;
    area := a * b * percent;
    mass := a * percent * b * c;
    rewrite(outFile,outFileName);
    writeln(outFile,'A = ',a:8:4);
   writeln(outFile,'B = ',b:8:4);
writeln(outFile,'C = ',c:8:4);
writeln(outFile,'Percent full = ',percent:8:4);
writeln(outFile,'Type Theta Moment');
    while (angle (= 90.0) do
    begin
        moment := 0.0;
        beta := ((90 - angle)/360)*(2 * pi);
        theta := (angle/360)*(2 * pi);
        tangentBeta := sin(beta)/cos(beta);
        tangentTheta := sin(theta)/cos(theta);
        form := WhatType;
        case (form) of
             1:TypeOne;
             2:TypeTwo;
             3:TypeThree;
             4:TypeFour;
             5:TypeFive;
             6:TypeSix;
            7:TypeSeven;
        writeln(outFile,form, ' ',angle:6:2, ' ',moment:8:4);
        angle := angle + increment;
    end;
    close(outFile);
End. (*END MAIN PROGRAM*)
```